From Double Pendulum to Lava Lamp: Linear Embeddings That Tame Chaotic Time Series

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

January 6, 2026

Chaos is often described as “randomness with rules,” but the rules can be invisible in the measurements we actually record. A double pendulum can be fully deterministic and still look unpredictable; a lava lamp can appear like pure disorder while quietly obeying constraints imposed by buoyancy, viscosity, and heat flow. Recent machine-learning methods suggest a disciplined way to bridge that gap: instead of trying to write a simple law in the original variables, we learn a new coordinate system in which the dynamics become approximately linear and low-dimensional. This paper uses two intuitive exemplars—an instrumented double pendulum and a video-recorded lava lamp—to explain what “linearization by learned coordinates” means, what it can and cannot promise, and how to evaluate it without falling for narrative temptation. The aim is not to claim that chaos collapses into a magical single equation, but to show how compact latent models can support forecasting, stability analysis, and interpretable mode discovery when the data are rich and the validation is honest. We close by proposing a practical experimental protocol, metrics, and failure tests that a reader can reuse across physical, biological, and engineered systems.

Keywords: chaos, double pendulum, lava lamp, low-dimensional embedding, Koopman operator, linear latent dynamics, autoencoder, time-delay embedding, dynamical systems, nonlinear forecasting, stability analysis, Lyapunov function, eigenmodes, system identification, spatiotemporal video dynamics, interpretability, model validation, physics-informed machine learning.

A collaboration with GPT-5.2-Thinking. 56 pages. CC4.0.

Outline

1. Introduction: When Chaos Looks Like Noise
1.1 The recurring error: mistaking unpredictability for randomness
1.2 Why “one equation” claims go viral
1.3 The disciplined alternative: change coordinates, then analyze
1.4 What this paper will and will not claim

2. Conceptual Core: Linear Dynamics in Learned Coordinates
2.1 The idea of a state and why measurement space is not state space
2.2 Time-delay reconstruction as a bridge from observations to state
2.3 Autoencoders as coordinate discoverers, not just compressors
2.4 Koopman-style linearization: what “linear” means here
2.5 Why the “simple equation” is usually a linear operator, not a scalar formula

3. Two Exemplars
3.1 Double pendulum as the canonical low-DOF chaotic system
3.2 Lava lamp as a high-dimensional, topological-change system
3.3 What makes one “hard” and the other “hard in a different way”

4. The Double Pendulum Case Study
4.1 Minimal sensor sets: angles, angular velocities, and timing
4.2 Building the time-delay vectors: choosing delays and horizons
4.3 Learning the latent model: encoder, linear step, decoder
4.4 What success looks like: forecasting and phase-portrait fidelity
4.5 What failure looks like: drift, mode collapse, and spurious stability
4.6 Why chaotic regimes often resist dimension reduction

5. The Lava Lamp Case Study
5.1 Treating video as a dynamical system
5.2 Representation choices: pixels, optical flow, segmentation, latent states
5.3 The merge/split problem: topological events and discontinuities
5.4 What a latent linear model can still provide
5.5 When the correct model is multi-regime or hybrid rather than single-regime
5.6 Practical shortcuts: extracting macrostates from embeddings

6. Evaluation: How Not to Fool Yourself
6.1 Forecasting metrics: one-step accuracy vs long-horizon fidelity
6.2 Geometry metrics: attractor structure, recurrence, and neighborhood preservation
6.3 Stability metrics: perturbation tests and empirical Lyapunov behavior
6.4 Baselines: naive persistence, local linear, and standard neural sequence models

6.5 Ablations: removing delays, reducing latent size, weakening constraints
6.6 Out-of-distribution tests: new energy levels, new temperatures, new lighting

7. Interpretation: What the Learned Linear Model “Means”
7.1 Eigenmodes as reusable features and coordinate axes
7.2 Pseudo-eigenfunctions in mixing/chaotic regimes
7.3 When interpretability is real vs when it is a story
7.4 Linking latent variables to physical quantities: cautious mapping strategies

8. What This Enables Beyond These Two Toys
8.1 Sensors and machines: vibration, bearings, and fault precursors
8.2 Circuits and power systems: oscillations, instabilities, regime shifts
8.3 Physiology and neuroscience: reduced dynamics without oversimplification
8.4 Climate and geophysical data: compression with accountability
8.5 Control prospects: using latent linearity for planning and stabilization

9. Limits, Risks, and Misreadings
9.1 Determinism is not simplicity
9.2 Stochastic drivers and hidden variables: when embedding can’t rescue you
9.3 Topological change and non-smooth events
9.4 Dataset temptation: the difference between fit and understanding
9.5 The rhetoric hazard: “chaos has a formula” as category error

10. A Reusable Protocol
10.1 Data collection checklists for low-DOF vs video systems
10.2 Training recipe: staged objectives and horizon growth
10.3 Reporting standards: plots, tables, and failure cases to always include
10.4 A minimal open benchmark suite for chaotic embeddings

11. Conclusion: Order Without Hype
11.1 What we can honestly claim
11.2 What remains open
11.3 Why this matters: analysis, prediction, and humility

References

Art

1. Introduction: When Chaos Looks Like Noise

Most people learn the word “chaos” as a synonym for disorder. In everyday life that works well: traffic is chaotic, politics is chaotic, the weather is chaotic. But in dynamical systems, chaos is not the absence of rules. It is what can happen when rules are present and deterministic, yet their consequences amplify small differences so quickly that prediction becomes fragile. The result is a familiar psychological trap: if we cannot reliably forecast, we start to speak as if the system itself is random. This paper begins from that trap, because it is exactly the trap that “chaos finally has a formula” headlines exploit.

Two concrete exemplars keep us honest. The double pendulum is a mechanical system with a small number of degrees of freedom whose equations of motion are well known. And yet its measured motion can look like pure unpredictability. The lava lamp is the opposite: an apparently simple object whose visual behavior is high-dimensional and punctuated by merge-and-split events that feel “beyond equations.” These two cases frame the central question of this paper: not whether chaos has rules, but whether there exist coordinate choices in which messy measurements become analyzable without turning analysis into mythology.

1.1 The recurring error: mistaking unpredictability for randomness

The error has a consistent structure. We observe a system, we attempt a forecast, the forecast degrades rapidly, and we conclude that the process is effectively random. But “effectively random to a particular observer with limited measurements” is not the same claim as “fundamentally random.” A double pendulum makes this difference painfully clear. If you track only one angle and ignore velocities, you throw away state. If you sample too slowly, you alias the dynamics. If your measurements are noisy, the inferred state jitters. Each of these produces the same phenomenology: your predictions fall apart quickly. None of them implies that the system lacks determinism.

The lava lamp makes a different version of the same point. You can record video at high frame rate and still fail to build a predictive model, because the “state” is not just the pixel array. The state includes temperature gradients, local viscosity, buoyancy-driven flow, and boundary conditions in the glass. Many of those are unobserved. Again, forecast failure does not automatically mean randomness. It can mean that the system is deterministic but partially observed, high-dimensional, or undergoing regime changes.

So the recurring error is not merely semantic. It is a methodological collapse: we treat prediction failure as metaphysical evidence. This paper treats it as an engineering signal: either our measurements do not capture state, our model class cannot express the dynamics, or our evaluation is too weak to detect self-deception.

1.2 Why “one equation” claims go viral

“One equation” is a perfect meme because it promises relief from cognitive load. It converts a world that feels unmanageable into a story with a single lever. It also flatters the reader: you can hold the whole thing in your head. The claim is especially seductive in chaos, where people already feel embarrassed by their inability to predict. A headline that says chaos has finally been compressed into a simple formula offers a kind of epistemic salvation.

But the meme works by conflating several different meanings of “equation.” In many modern machine-learning approaches to dynamics, the visible “simple equation” is not a law in the original variables. It is a law in a learned latent coordinate system. The full learned system typically contains a nonlinear encoder, a compact latent update rule, and a nonlinear decoder. If you only show the middle step, it looks like miraculous compression. If you show the full pipeline, it looks like what it is: a learned change of coordinates plus a linear surrogate model.

The other reason these claims spread is that they exploit a real truth: chaos is deterministic in many important systems. So the viral story has a kernel of legitimacy. The problem is the slide from “deterministic dynamics can look random” to “randomness was fake and simplicity was hidden all along.” That slide is not a small exaggeration. It changes what the work means and what it can be used for.

1.3 The disciplined alternative: change coordinates, then analyze

This paper’s alternative is old in spirit and new in instrumentation. Old, because the idea that the right variables make a problem solvable is as ancient as physics itself. New, because we now have practical tools that can search for those variables automatically from data, using representation learning.

The disciplined approach has three commitments.

First, we separate measurement space from state space. What we measure (angles, pixels, sensor voltages) may be an awkward projection of the true state. When that happens, we can sometimes reconstruct a usable state by incorporating recent history (time delays) and learning coordinates that behave more like state variables.

Second, we aim for linearity in the learned coordinates not because the world is linear, but because linear models are analyzable. If a nonlinear system can be represented as approximately linear evolution in a latent space, we gain access to tools that quantify growth and decay of modes, stability tendencies, and long-horizon predictability limits.

Third, we insist on evaluation that can catch storytelling. A model that produces a pretty short-term fit can still be useless. The test is not whether a plot looks smooth; it is whether the

learned coordinates and latent evolution remain consistent under perturbations, new initial conditions, and modest shifts in operating regime.

Applied to the double pendulum, this means asking whether the learned latent system preserves the geometry of trajectories and respects the system's sensitivity without inventing false stability. Applied to the lava lamp, it means asking whether the latent variables track reproducible macrostates and whether the model's failures correlate with real physical regime shifts (warm-up, cooling, blob merge/split events) rather than arbitrary drift.

1.4 What this paper will and will not claim

This paper will claim four things.

First, chaos can look like noise because our measurements are often the wrong coordinates for analysis. Second, representation learning can sometimes discover coordinates in which dynamics become simpler and more linear, enabling forecasting and stability-oriented summaries that are otherwise hard to obtain from raw measurements. Third, the double pendulum and the lava lamp form a useful pair of testbeds because they stress different failure modes: low-dimensional chaos with strong sensitivity versus high-dimensional visual dynamics with discontinuous events. Fourth, the right way to talk about these methods is as disciplined system identification, not as the discovery of a magical universal equation.

This paper will not claim that chaos is "solved," that weather becomes easy, or that a single human-readable formula replaces the need for measurement, modeling, and validation. It will not claim that a learned latent linear model reveals the "true" laws of a system in any metaphysical sense. And it will not treat compression as understanding. The goal is narrower and more useful: a workflow that can produce compact, testable, analyzable representations of complex time series, while making it difficult for the author—and the reader—to confuse a good story with a good model.

2. Conceptual Core: Linear Dynamics in Learned Coordinates

The central idea of this paper is simple to say and easy to misunderstand: many systems that look chaotic or "too complex" in their measured variables can look far more structured after a change of coordinates. The change of coordinates is not chosen by hand from physical insight alone. It is learned from data, with the explicit goal that the system's evolution in the new coordinates is approximately linear. Once you have a linear model, you inherit a century of tools for analyzing growth, decay, stability, modes, and prediction horizons. The difficulty is not in writing the linear update rule. The difficulty is in learning coordinates where that rule is even

approximately true, and then proving to yourself that you have learned something real rather than a flattering compression artifact.

This section lays down the conceptual spine needed to evaluate the two exemplars later (double pendulum and lava lamp). The key move is to keep three spaces distinct: the physical state space of the system, the measurement space you actually observe, and the learned latent space in which you try to linearize dynamics.

2.1 The idea of a state and why measurement space is not state space

A “state” is whatever information is sufficient to predict future evolution, given the system’s rules and external inputs. If you know the true state at time t, you do not need the past: the future is determined (or at least statistically determined, if the system has noise) by the state and the inputs. This is the meaning of “Markov” in dynamical modeling: the next step depends on the current state, not on a long history.

The problem is that we rarely measure the state. We measure a projection. In a double pendulum, the true mechanical state might be described by angles and angular velocities (and perhaps frictional states, compliance, and sensor biases). If you measure only one angle, you do not have a state. If you measure both angles but not velocities, you still do not have a state. Your measurements can look erratic precisely because you are missing information that is dynamically essential.

In a lava lamp, the mismatch is more extreme. The “state” would include temperature fields, flow fields, local densities, viscosities, and boundary conditions, across a continuous volume. A camera gives you a 2D projection, with lighting artifacts, occlusions, and limited frame rate. Here, measurement space (pixels) is not only a projection of state; it is a nonlinear, lossy rendering of an already high-dimensional field.

So the first conceptual discipline is to stop treating the measured variables as privileged. They are convenient, not fundamental. If your goal is analysis and prediction, you want to recover a state-like representation, even if that representation is not directly interpretable as a physical variable.

2.2 Time-delay reconstruction as a bridge from observations to state

If you cannot measure the full state, one of the oldest tricks in the book is to use time. A single measurement at one instant can be an impoverished projection. But a short history of measurements can contain enough information to infer the missing degrees of freedom.

Time-delay reconstruction embodies that intuition. You create a vector that stacks the observation at time t together with the observation at times t - tau, t - 2tau, and so on. The delays supply context: if you see how the observation has been changing, you can infer

something about the hidden velocities, phases, or internal variables that are not directly measured.

This is not magic. It is a structured way of manufacturing a higher-dimensional representation from a low-dimensional sensor. In a double pendulum, if you only measure angles, time delays can partially supply velocity information. In a lava lamp video, time delays supply motion cues, implicitly approximating optical flow and temporal continuity. In both cases, the delays move you closer to a representation in which the dynamics become “more Markov”: the next step becomes more predictable from the constructed vector.

Two cautions matter.

First, delay reconstruction assumes that the system is not being driven by wildly changing unobserved inputs. If an external driver changes unpredictably, no amount of delays will reconstruct it unless it leaves a signature in the observations.

Second, delay reconstruction introduces design choices: how long a delay tau should be, how many delays to stack, and what sampling rate is adequate. These choices are not cosmetic. Too small a tau makes successive delayed observations nearly redundant. Too large a tau can skip over important dynamics and entangle unrelated phases.

The practical payoff is this: delays give you a richer input space on which learning can operate. They do not solve the modeling problem by themselves, but they reduce the burden placed on the model to infer state from a single snapshot.

2.3 Autoencoders as coordinate discoverers, not just compressors

Autoencoders are often described as compression: map data into a small latent vector, then reconstruct it. That framing is incomplete for dynamical systems. Here, the goal is not only to compress, but to discover coordinates that behave like state variables.

In this setting, the encoder is a learned coordinate map. It takes a measurement-derived representation (often a time-delay vector) and outputs latent coordinates z. Those coordinates are not required to resemble the original variables. In fact, it is often better if they do not. The job of z is to be a space in which evolution is simple.

The decoder plays an equally important role. It is the bridge back to the measurement world. If the decoder can reconstruct what you observe, then z has not thrown away information that matters for your purposes. But reconstruction alone is not sufficient. A latent space can reconstruct well and still be dynamically useless. It can be a clever codebook rather than a state.

This is why the dynamical constraint matters: the latent representation is trained not only to reconstruct, but to evolve in a predictable, structured way under a simple rule.

For the double pendulum, the latent coordinates might discover combinations that align with dominant modes of motion or energy exchange. For the lava lamp, the latent coordinates might discover macrostates corresponding to blob counts, rise rates, or heating regimes—things that are not explicitly labeled but are consistently present in the data.

The key point is conceptual: the autoencoder is not primarily a data compressor. It is a coordinate system learner, and its value is measured by the quality of dynamics you can express in its coordinates.

2.4 Koopman-style linearization: what "linear" means here

The promise of Koopman-style thinking is that nonlinear dynamics can be represented linearly if you choose the right functions of the state. In classical terms, a nonlinear state x_t evolves by a nonlinear rule. But if you lift x_t into a space of observables (functions of x_t), then the evolution of those observables can be expressed by a linear operator.

In practice, no one has access to the infinite-dimensional Koopman operator. The art is to find a finite-dimensional approximation that is useful. The machine-learning strategy is to learn observables (the latent coordinates) that make the finite-dimensional approximation as accurate as possible over the time horizons you care about.

So "linear" in this paper does not mean "the world is linear." It means:

- there exists a latent coordinate map z = psi(x) such that
- z evolves approximately as z_{t+1} = A z_t (discrete time) or dz/dt = G z (continuous time),
- and the original observations can be recovered approximately by x ≈ phi(z) for some decoder phi.

This kind of linearity is powerful because it turns prediction into repeated multiplication by a matrix (or exponentiation of a generator). It also turns analysis into spectral questions: eigenvalues, eigenvectors, growth rates, and invariant subspaces.

But it is also dangerous, because "approximately linear" is a sliding concept. A linear model can be locally accurate and globally wrong. It can work on the training regime and fail under small shifts. It can appear stable because the latent space has learned to hide instability in the decoder. These are not hypothetical failure modes; they are common in representation learning when evaluation is weak.

So the meaning of "linear" must remain operational: it is linear to the extent that the latent evolution predicts and explains behavior under tests that the author did not design to be easy.

2.5 Why the “simple equation” is usually a linear operator, not a scalar formula

The viral claim “compressed into one simple equation” invites the reader to imagine a single compact expression in the original variables, like a law of motion written on a napkin. That is rarely what these methods produce.

What they usually produce is a linear operator: a matrix A (or generator G) that advances a latent vector z. Even if you write it as one equation, z_{t+1} = A z_t, the substance is in A. A is not a scalar; it is a structured object containing many parameters. Its size is the latent dimension squared, and its eigenstructure matters more than its individual entries.

Moreover, the “one equation” is not the whole model. The encoder and decoder are nonlinear maps learned by a neural network. If you are honest, the model is a three-part system:

- encode: X (measurements or delays) -> z
- evolve: z -> A z (or e^{G dt} z)
- decode: z -> reconstructed measurements

The linear middle step is the part that looks like a tidy formula. But the learned coordinate maps are where the “difficulty” has been moved. They are the price you pay to earn linearity.

This is why the best way to read a “one equation” claim is as a claim about representation, not about ontology. The method does not reveal that the world is simple. It reveals that a complex measured process may have a representation in which a low-dimensional linear surrogate is useful for prediction and analysis. That is a practical and important result. It is also far from the metaphysical claim that chaos has been “solved.”

In the rest of the paper, the double pendulum and lava lamp will serve as stress tests for this conceptual core. One will show you how far linearization can go even in a low-dimensional chaotic system. The other will show you what happens when the system is high-dimensional, visually rendered, and punctuated by events that violate the smooth assumptions implicit in many latent linear models.

3. Two Exemplars

To keep the discussion concrete, this paper uses two physical exemplars that sit on opposite ends of the modeling spectrum. The double pendulum is mechanically simple in construction and low-dimensional in principle, yet famously unpredictable in practice. The lava lamp is visually simple in concept—warm wax rises, cool wax falls—yet the observed behavior is high-dimensional, visually rendered, and punctuated by events that violate the smoothness

assumptions of many standard models. Together they form a useful pair because they force us to face two distinct reasons “chaos looks like noise”: sensitivity in a low-degree-of-freedom system, and partial observability plus topological change in a high-dimensional system.

The goal is not to elevate either system to a grand metaphor. The goal is to use them as honest testbeds for the claim that learned coordinates can yield a latent space in which dynamics are approximately linear, analyzable, and forecastable. If the method cannot behave sensibly on these two systems, it is unlikely to behave sensibly on harder domains like weather, physiology, or complex machinery.

3.1 Double pendulum as the canonical low-DOF chaotic system

A double pendulum has an almost unfair pedagogical power: it demonstrates that you can have a deterministic system with a small number of degrees of freedom and still lose practical predictability. In its idealized form, the state can be described by a handful of variables: two angles, two angular velocities, and parameters describing the lengths and masses (plus friction if you model dissipation). On paper, it is “just mechanics.”

In the lab, it becomes a chaos machine. Two runs starting from nearly identical initial conditions can diverge rapidly in phase. Any slight difference in release angle, friction at the pivot, air currents, or sensor noise becomes magnified. This is what people mean when they say chaos has sensitive dependence: it does not require randomness; it requires only amplification.

This makes the double pendulum a canonical test for any method claiming to “tame chaos.” The system is deterministic enough that we can attribute most unpredictability to sensitivity and measurement limitations, not to intrinsic randomness. It is low-dimensional enough that we cannot hide behind the excuse that “there are too many variables.” And it is rich enough dynamically—periodic motion, quasiperiodicity, chaotic tumbling—that a model must handle multiple behaviors.

For learned-coordinate approaches, the double pendulum raises a central question: can we find a representation in which the evolution is simple without denying the system’s true instability? A good method should not magically turn chaos into a stable spiral. It should represent the instability honestly while still extracting analyzable structure: dominant modes, characteristic timescales, and stable versus unstable manifolds where appropriate.

3.2 Lava lamp as a high-dimensional, topological-change system

A lava lamp offers the opposite kind of challenge. In the mind’s eye it seems trivial: buoyancy and heat create blobs that rise and fall. But the observation we actually use—video—is a complex rendering of a complex fluid system.

From the modeling perspective, a lava lamp is “hard” in at least four ways.

First, the underlying physical system is high-dimensional. It is not a few coordinates; it is a set of coupled fields: temperature, flow velocity, density, and phase distribution in a container with boundaries. Any finite sensor set is a drastic compression.

Second, our primary measurement is typically visual. Pixels are not state variables. They are a projection influenced by lighting, reflections, camera angle, lens distortion, and occlusion. Two frames can look different because the wax changed, or because the camera’s auto-exposure changed.

Third, the system has regime structure. Warm-up dynamics differ from steady-state dynamics. The lamp can pass through phases where blobs nucleate, stretch into tendrils, merge, split, and dissolve. A single global model may be inadequate; a better description may be multi-regime or hybrid.

Fourth, and most crucially, the system contains topological change. A blob divides into two blobs; two blobs merge; a blob forms a thin neck and pinches off. These events are not smooth in any naïve “object identity” representation. Even if the physics is smooth at the continuum level, the perceptual objects we track are not. Many models fail precisely at these event boundaries, because they were trained to represent continuous deformation rather than identity changes.

For learned-coordinate approaches, the lava lamp is a stress test of a different sort: can a latent linear model survive when the measurement is high-dimensional and eventful? Can it learn macrostates that remain stable across lighting changes and viewpoint shifts? Can it predict short-horizon motion while admitting that merges and splits are fundamentally harder? Most importantly, can we construct evaluation that reveals where the model works and where it breaks, rather than averaging both into a misleading global score?

3.3 What makes one “hard” and the other “hard in a different way”

Both systems can be described as chaotic in a colloquial sense, but the reasons are distinct.

The double pendulum is hard because it is a low-dimensional deterministic system with strong sensitivity. The main obstacles are:

- state inference from imperfect measurements
- rapid divergence of trajectories under tiny perturbations
- separating measurement noise from genuine instability
- modeling dissipation and friction realistically enough to avoid drift artifacts

In other words, the pendulum's difficulty is "intrinsic" to the dynamical sensitivity of the system given limited measurement precision. Even if you know the equations, your ability to predict is bounded by your ability to specify the initial state and parameters.

The lava lamp is hard because it is high-dimensional, partially observed, and full of event boundaries. The main obstacles are:

- the measurement is a lossy rendering, not a state
- the system's relevant variables are distributed fields, not a few coordinates
- object identities appear and disappear through merge/split events
- the dynamics are often multi-regime and nonstationary (warm-up, steady state, cooling)
- externalities like lighting and camera settings can masquerade as dynamics

In other words, the lamp's difficulty is "extrinsic" to observation and representation: even if the physics is lawful, the measurement channel and the chosen representation can make the dynamics look discontinuous and arbitrary.

This distinction matters because it prevents a category error. If a learned-coordinate method struggles with the double pendulum, the problem is likely that chaos resists low-dimensional linearization in a global sense, or that the method is too eager to pretend stability. If the method struggles with the lava lamp, the problem may be that the observation is not state-like, that topological events violate smooth assumptions, or that we need a different architecture (for example, one that explicitly models events and regimes rather than insisting on a single linear latent flow).

So the two exemplars are not chosen as cute visuals. They are chosen because they force the method to confront two different kinds of hardness, and because they force the author to keep separate the questions "is the system deterministic?" and "do our measurements and representations allow a useful state model?" That separation is the beginning of epistemic discipline, and it is the only reliable antidote to viral "one equation" narratives.

4. The Double Pendulum Case Study

The double pendulum is the cleanest place to learn the difference between "a system is deterministic" and "a system is predictable." It is a low-degree-of-freedom mechanical device whose motion can be written down in closed form (in principle), yet it produces trajectories that diverge rapidly under small perturbations. That combination makes it the right first case study for learned-coordinate linearization. If a method claims it can extract a linear latent model from

"chaos," the double pendulum forces the method to face real sensitivity while keeping the system small enough that we cannot excuse failures as "too many variables."

In this section we treat the double pendulum as a data-generating instrument. We define a minimal measurement setup, construct time-delay vectors that are closer to state, learn a latent linear dynamical model, and then evaluate it with tests designed to detect self-deception. The core stance is pragmatic: the question is not "can we perfectly predict a chaotic system?" but "can we learn a representation that supports honest forecasting and analysis over a useful horizon, and can we prove where it breaks?"

4.1 Minimal sensor sets: angles, angular velocities, and timing

A "minimal" sensor set is one that (a) contains enough information to approximate state, and (b) is realistic for a reader to implement without specialized lab infrastructure.

For a double pendulum, the minimal state-like measurement is:

1. theta_1(t), theta_2(t) (joint angles)
2. omega_1(t), omega_2(t) (joint angular velocities)
3. a consistent timestamp t

There are three practical ways to obtain this.

First, rotary encoders at the pivots. This is the cleanest for angles and timing. Velocities can be computed by differentiation, but raw numerical derivatives amplify noise, so some form of smoothing is almost always required.

Second, video tracking. This is accessible to most readers, but it introduces projection errors, lens distortion, occlusion, and frame-rate limits. It can still work well if you track two arm endpoints and solve for angles geometrically. Velocities again require careful differentiation.

Third, inertial sensors on each arm. This can be surprisingly effective, but it adds calibration complexity and can drift.

Regardless of the sensing modality, two measurement disciplines matter more than the sensor brand.

Sampling rate: chaotic divergence is not helped by undersampling. If the sampling is too slow, the model sees a blurred version of the dynamics and learns spurious smoothness. In practice you want a sampling rate high enough that the motion between samples is small relative to the curvature of the trajectory. If you cannot sample faster, you should treat the model as a coarse-grained dynamical system and evaluate it accordingly.

Timebase quality: irregular timestamps and dropped frames can quietly destroy training. If your measurement is asynchronous or jittery, you must either resample to a uniform grid or incorporate delta_t explicitly into the model.

Velocity estimation: omega(t) is not a "free" measurement if you derive it. The easiest trap is to compute omega by finite differences and feed it directly. The resulting noise can dominate the latent learning. A safer workflow is to estimate omega with a smoothing-and-differentiation procedure that preserves phase and does not inject artificial lag. If you cannot obtain reliable omega, time-delay vectors (Section 4.2) become even more important because they allow the model to infer velocity-like information from angle history.

Finally, note what minimal does not mean. It does not mean frictionless. Real pendulums have dissipation, pivot friction, and sometimes slight compliance. If you ignore dissipation entirely, the model may compensate by inventing latent decay terms that look like "stability" but are merely unmodeled friction.

4.2 Building the time-delay vectors: choosing delays and horizons

The central modeling problem is that measurements are not automatically a Markov state. Even if you measure angles and velocities, you may still miss slow variables: friction state, temperature effects, bias drift, or tiny asymmetries. Time-delay reconstruction is a practical bridge: it manufactures a richer "state proxy" by stacking a short history of observations.

Let y(t) be the measurement vector at time t. A time-delay vector at time t is:

Y_t = [y(t), y(t - tau), y(t - 2*tau), ..., y(t - (m-1)*tau)]

Here tau is a delay step and m is the number of delays.

Choosing tau: too small makes the delays redundant; too large skips essential curvature. A practical heuristic is to pick tau such that y(t) and y(t - tau) are not trivially correlated but still clearly related. If you have enough data, you can choose tau by an information or autocorrelation criterion, but the key is conceptual: tau should reveal motion, not repeat the same frame.

Choosing m: larger m gives more history but increases input dimension and can make learning harder. For a double pendulum, m is often a lever for how much velocity and phase information the model must infer. If you cannot measure omega well, increase m modestly and let the model infer velocity from angle history. If you have good omega, smaller m can suffice.

Choosing the prediction horizon: there are two horizons that matter and they are not the same.

1. Training horizon H_train: how many steps forward the loss function asks the model to predict.

2. Evaluation horizon H_eval: how far forward you actually test before declaring failure.

A common failure mode is to train on H_train = 1 (one-step prediction) and then declare success by showing a short roll-out. Chaotic systems punish that. The model can be excellent at one-step prediction and still drift into nonsense over 50 steps. A disciplined workflow trains with a growing multi-step horizon and evaluates on longer roll-outs than the model saw during training.

A final practical point: normalization. Angle variables wrap (pi discontinuities), and naive normalization can introduce jumps. A robust approach is to represent angles by sine/cosine pairs or unwrap angles before building delays. The goal is to prevent the model from learning the artificial discontinuity at +/-pi as if it were a physical event.

4.3 Learning the latent model: encoder, linear step, decoder

The learned-coordinate approach has a three-block structure:

1. Encoder: z_t = f_enc(Y_t)
2. Latent evolution: z_{t+1} = A z_t (discrete) or z(t + dt) = exp(G*dt) z(t) (continuous-time form)
3. Decoder: y_hat(t) = f_dec(z_t)

The "simple equation" that shows up in viral summaries is the latent evolution step. But the representational work is done by the encoder and decoder: they discover coordinates z in which a linear operator A (or generator G) is a good approximation over the horizons that matter.

Training objectives typically include three coupled goals.

Reconstruction: y_hat(t) should match y(t). This prevents z from discarding information needed to represent the observed motion.

Dynamics consistency: the decoded roll-out from latent evolution should match future measurements. This is the heart of learning a state representation rather than a static compression.

Multi-step stability: the roll-out should remain consistent not just for one step but across multiple steps, because chaos makes short-horizon success cheap.

A practical training schedule for chaotic data is staged.

Stage 1: train the autoencoder to reconstruct. This stabilizes the latent space and prevents early training from collapsing into trivial solutions.

Stage 2: introduce the linear latent evolution term and enforce one-step prediction.

Stage 3: increase the prediction horizon gradually (for example, 1 step, then 5, then 10, then 25) while rebalancing the loss weights so the model cannot "win" by reconstructing well but predicting poorly.

This staged approach matters because a model that is forced into long-horizon prediction too early may learn to damp everything into a stable point (an easy way to reduce long-horizon error if the loss is not designed carefully). Which leads directly to the next sections: what counts as success, and what failure looks like when it is disguised as success.

4.4 What success looks like: forecasting and phase-portrait fidelity

In a chaotic system, "success" must be defined with care. Perfect long-horizon trajectory matching is not the right bar. Small errors in inferred state should grow. So we define success in layered, operational terms.

Short-horizon forecasting: the model should predict the next few steps accurately, and accuracy should degrade in a manner consistent with sensitivity rather than sudden unphysical collapse. The goal is not infinite forecast length; the goal is honest forecast decay.

Medium-horizon statistical fidelity: even when pointwise trajectories diverge, the model should preserve distributional and geometric properties. For the double pendulum, that means the model's roll-outs should populate the same regions of state space and exhibit similar recurrence behavior.

Phase-portrait fidelity: if you plot trajectories in (theta_1, omega_1) and (theta_2, omega_2) planes (or other canonical projections), the learned model should reproduce the qualitative structure: lobes, turning regions, and the density of visits. This is where "looks chaotic" is not enough; the geometry has to match.

Constraint-aware behavior: real pendulums lose energy due to friction. A successful learned model should reflect dissipation in a controlled way: the decay should be gradual and consistent with the observed system, not an artifact that appears only during roll-out.

Robustness across initial conditions: success cannot be a single demo trajectory. The model should generalize across many starts (including different release angles and energy levels), and its failure modes should be consistent and explainable.

Most importantly, success includes the ability to say "this is where the model stops being predictive." A model that cannot estimate its own horizon is not a scientific instrument; it is a storyteller.

4.5 What failure looks like: drift, mode collapse, and spurious stability

Chaotic modeling is full of failures that look like wins. Here are the most common ones in learned-coordinate linearization.

Drift: the roll-out slowly departs the observed manifold and never comes back. In plots this often looks like a trajectory that "wanders off" into regions never seen in real data. Drift can come from slight bias in the latent linear operator that accumulates with each step.

Mode collapse: the latent space loses degrees of freedom and maps many distinct states to a small region. Reconstruction may remain decent, but dynamics degrade because the model no longer distinguishes states that evolve differently. In a chaotic system this can produce artificial synchronization: different initial conditions collapse to the same latent path.

Spurious stability: the model learns a latent operator with eigenvalues that imply damping, pulling trajectories toward a fixed point or a limit cycle even when the real system remains chaotic at that energy level. This is a classic failure because damping is an easy way to reduce multi-step prediction loss. If the training objective does not penalize this properly, the model will choose fake stability.

Decoder cheating: the latent evolution is smooth and stable, but the decoder is highly nonlinear and turns small latent changes into complex-looking outputs. This can produce plots that look "chaotic" while the latent dynamics are actually a simple attractor. It is chaos as a rendering trick.

Overfitting to a regime: the model performs well for one energy band or one set of release angles but fails abruptly for others. This is especially common if the dataset was collected in a narrow operating range.

Timebase mismatch: the model implicitly assumes a fixed dt but the data have jitter. The latent operator then absorbs timing artifacts, and the model fails when resampled or when dt changes.

A disciplined evaluation does not merely report average error. It includes failure demonstrations: where drift begins, how spurious stability appears, and which regimes are out of scope. In this paper, failure is treated as data: it tells you whether the learned coordinates are state-like or merely compressive.

4.6 Why chaotic regimes often resist dimension reduction

The most important correction to the viral narrative is this: chaos does not automatically become low-dimensional simply because it is deterministic. A system can have a small number of degrees of freedom and still require a rich representation to model globally.

There are several reasons.

Continuous spectral structure: strongly chaotic systems behave as if they have a broad spectrum of interacting modes. If you try to represent them with a tiny set of linear modes, you often fail because no small set of eigenmodes captures mixing behavior globally. The best you can do may be local or regime-specific linearization.

Attractor dimension is not always tiny: while some chaotic attractors are low-dimensional, others are not "compressible" to a handful of coordinates without losing essential dynamics. Even if the physical system is low-DOF, dissipation, frictional effects, and measurement artifacts can effectively raise the dimension required for accurate state representation.

Partial observability: if your measurements do not capture state cleanly, the model needs extra latent dimensions to represent hidden variables or to compensate for observation noise. Ironically, worse sensors can force higher latent dimension.

Nonstationarity: if friction changes with time, pivots warm up, or the apparatus slightly shifts, the system is not one dynamical system but a family of nearby systems. A low-dimensional model that assumes stationarity will either drift or invent damping to cope.

Global vs local linearization: many systems are locally linearizable but not globally linearizable by a single low-dimensional linear operator. A single A matrix may be an oversimplification; you may need multiple operators (regimes) or a latent space large enough to approximate global behavior.

So, for the double pendulum, it is not a failure if the best-performing latent model does not reduce dimension dramatically. That outcome can be the most honest scientific result: the method may still provide a usable linear surrogate for forecasting and analysis in a latent space of comparable dimension, while revealing that "one simple equation" is a rhetorical flourish, not an empirical conclusion.

This is why the double pendulum belongs early in the paper. It teaches a critical humility: the win is not compression. The win is a representation that supports prediction and analysis without lying about instability.

5. The Lava Lamp Case Study

If the double pendulum is the clean stress test for low-dimensional chaos, the lava lamp is the clean stress test for high-dimensional observation. Almost everyone has watched a lava lamp long enough to feel the “this is pure chaos” intuition. The wax blobs form, rise, stretch, merge, split, and dissolve. You can predict the rough direction of motion (warm rises, cool sinks) but

you cannot predict the exact shapes and timings for very long. That gap between qualitative predictability and quantitative unpredictability is exactly where learned-coordinate methods are most tempting and most dangerous.

The lava lamp is also pedagogically honest because it forces the reader to confront a central theme of this paper: measurement space is not state space. Video is not a direct physical observable like angle or voltage. It is a rendering. The challenge is therefore not only to learn a latent linear evolution, but to decide what you mean by "the system" in the first place. Are you modeling the fluid physics? The blob dynamics? Or the pixel movie produced by a camera in a room? Those are not the same object, and success in one sense can be failure in another.

This section treats the lava lamp as a video dynamical system. The practical goal is not to solve fluid dynamics. The goal is to extract a compact representation that supports useful forecasts, regime detection, and macrostate summaries while remaining honest about discontinuities and nonstationarity.

5.1 Treating video as a dynamical system

A video can be treated as a time series x(t) where each x is a high-dimensional vector (all pixel intensities). In that sense it is "just data," and the same machinery used for other time series can be applied: time-delay vectors, learned embeddings, and latent evolution models.

But the moment you do this, you inherit three traps.

First, the camera is part of the system. Auto-exposure, auto-white-balance, focus hunting, compression artifacts, and even tiny camera motion can create apparent dynamics that have nothing to do with the lamp. If the model learns those artifacts, it will appear to predict well on the same recording and fail catastrophically when lighting changes.

Second, video introduces observation nonlinearity. The mapping from physical state to pixels is nonlinear and lossy: occlusion, specular reflections, refraction through glass, and depth ambiguity are built in. Two different physical states can produce similar frames. That makes "state reconstruction" harder than in the pendulum.

Third, video is typically underspecified without timing discipline. Frame rates vary, frames drop, and timestamps are often irregular. A latent evolution model that assumes a constant dt will absorb timing noise into the learned operator.

Therefore, treating video as a dynamical system begins with defining a controlled data protocol, even if the hardware is simple:

- lock camera position and focus
- fix exposure/white-balance if possible

- maintain constant lighting, or record lighting conditions explicitly
- record stable timestamps and a stable frame rate, or resample later
- capture long sequences that include warm-up, steady state, and transitions

This is not perfectionism. It is how you prevent the model from learning the camera instead of the lamp.

5.2 Representation choices: pixels, optical flow, segmentation, latent states

There is no single “right” representation for lava lamp dynamics because the question “what is the system?” is already ambiguous. Different representations encode different assumptions about what you want to preserve.

Pixels: The simplest and most brute-force approach is to treat raw frames as the observations. This has two advantages: you don’t throw away information by preprocessing, and you avoid building a brittle pipeline that fails when segmentation fails. The downside is severe: the representation is enormous, sensitive to lighting, and filled with nuisance variation (reflections, background, noise). A model that learns from pixels alone must implicitly learn invariances that humans take for granted.

Optical flow: Flow encodes apparent motion between frames and is closer to “dynamics” than raw appearance. For a lava lamp, flow highlights rising blobs and falling streams. It can improve short-horizon prediction because it carries velocity-like information. But flow is itself an estimation algorithm with failure modes (aperture problems, occlusions, low-texture regions). You can end up modeling the flow estimator rather than the lamp.

Segmentation and object tracking: If you can segment wax from liquid and track blob components, you can represent the lamp as a set of objects with sizes, centroids, velocities, and shape descriptors. This is attractive because it reduces dimensionality drastically and creates interpretable variables. It is also brittle: segmentation quality varies with lighting and refraction, and tracking breaks precisely when blobs merge or split, which is when the dynamics are most interesting.

Latent states: Learned embeddings are the compromise: you use a neural encoder to map frames (or frame stacks) to a latent z that is much smaller than pixels but richer than a handful of hand-engineered features. This is aligned with the paper’s conceptual core: learn coordinates in which evolution is simple. The latent can be trained to ignore nuisances like background and lighting if the training objective and data augmentations force invariance.

In practice, the best approach often combines representations:

- use pixels or lightly processed frames as primary input (to avoid brittle segmentation)

- stack short frame histories (time delays) to make motion observable
- optionally include flow as an auxiliary channel or a regularizer
- use a learned latent as the modeling state

The representation decision is therefore not an implementation detail. It is an epistemic decision: what structure do you assume is preserved and what nuisances do you attempt to remove?

5.3 The merge/split problem: topological events and discontinuities

The lava lamp's signature behavior is not smooth deformation; it is topological change. A blob that was "one object" becomes "two objects." Two objects become one. Thin necks form and pinch off. These are discontinuities in any representation that depends on object identity.

This is the single biggest difference from the double pendulum. The pendulum's state evolves continuously in time (even if sensitive). The lava lamp's perceptual objects do not. Even if the underlying fluid equations are continuous, the human-level entities we track undergo discrete events.

Topological events create two modeling problems.

First, continuity assumptions fail. A latent linear model assumes that a single set of coordinates evolves smoothly under a linear operator. Merge/split events can cause sudden jumps in the appropriate representation. If the latent is forced to remain smooth, the model will smear the event, predicting a physically implausible in-between state.

Second, labeling becomes unstable. If you train with object-level representations, you must decide what happens to identities at merge/split. Many tracking algorithms will either arbitrarily assign identities or lose them. That injects non-physical noise into the time series.

There are three honest ways to handle this.

Option A: avoid object identity entirely. Use representations that do not require tracking: global shape descriptors, density fields, or latent embeddings learned from raw frames. Then merge/split is not a discrete identity event; it is a continuous change in the observed field. This sacrifices some interpretability but gains stability.

Option B: model events explicitly. Treat merges and splits as discrete events in a hybrid system: continuous dynamics within regimes, discrete transitions at event times. This is conceptually correct but more complex.

Option C: accept that some events are not predictable in detail and focus on predicting macrostates and statistics rather than exact configurations. This is the most practical scientific

posture: it mirrors how weather forecasting treats convection and turbulence at scales below model resolution.

5.4 What a latent linear model can still provide

Given these difficulties, what can a latent linear model honestly provide for a lava lamp?

Short-horizon prediction of motion patterns: Even if merge/split events are hard, many seconds of motion can be predicted in a coarse sense: blobs rise, stretch, drift, and fall. A latent linear evolution can capture dominant modes like “upward buoyant motion” versus “downward cooling return flow.”

Mode discovery: Linear latent dynamics yield eigenmodes and growth/decay rates. For a lava lamp, these modes may correspond to global circulation patterns, oscillatory heating cycles, or “activity level” fluctuations. Even if you cannot map each latent coordinate to a physical variable, the modes can be stable features for summarizing behavior.

Regime detection: A linear latent model often exposes when it stops being valid. Latent prediction error spikes can correlate with regime transitions: warm-up completion, onset of steady bubbling, or event-heavy periods. That can be useful even when exact video prediction fails.

Compression with dynamical meaning: Many video embeddings compress without preserving temporal structure. A dynamical latent embedding compresses while maintaining predictive content. That can support indexing, retrieval (“show me times when the lamp was in regime X”), and comparative analysis across lamps or temperature settings.

Stability-like summaries: “Stability” here means not the lamp settling into a static state, but the latent system’s tendency to remain within a bounded behavioral envelope. You can ask whether the lamp’s behavior is regular, strongly fluctuating, or drifting in activity over time.

The key point is that usefulness does not require perfect pixel prediction. A latent dynamical model can be valuable as a scientific instrument for segmentation-free analysis, provided it is evaluated with the right metrics.

5.5 When the correct model is multi-regime or hybrid rather than single-regime

A single global latent linear operator is an attractive fantasy. The lava lamp is a reminder that many real systems are better described as a family of local models with transitions between them.

There are at least three reasons multi-regime modeling is more correct here.

Warm-up vs steady state: The lamp begins far from equilibrium and approaches a quasi-steady behavior. The effective dynamics change as temperature fields stabilize. A single operator that tries to cover both will often average them into a poor compromise.

Event-driven transitions: Merge/split events change the effective degrees of freedom. After a large merge, the system may settle into a different flow pattern. A hybrid model that allows discrete transitions can represent this more faithfully.

External perturbations: Small changes in room temperature, lamp power, or airflow can shift behavior. These shifts act like slowly varying parameters, producing nonstationary dynamics.

A practical approach is to treat the latent linear model as local: it works within a regime, and you detect when the regime changes. The model can be implemented as:

- a mixture of linear operators in latent space
- a switching model where a discrete state selects which operator governs evolution
- a continuous-time model where the operator depends smoothly on a slow latent parameter
- an event-triggered model where large reconstruction or prediction residuals indicate a transition

The key is not to overcomplicate. The aim is to avoid forcing one elegant equation to do the job of a piecewise description that the system's behavior demands.

5.6 Practical shortcuts: extracting macrostates from embeddings

A reader may not want to build a full hybrid dynamical model. They may want a practical outcome: a small set of macrostates that summarize the lamp's behavior and a way to detect when the lamp changes state.

Embeddings make that possible with modest effort.

Workflow A: embedding then clustering

1. encode each frame (or short stack of frames) into a latent z
2. smooth z over time to reduce camera noise
3. cluster z into K states (K chosen by validation and interpretability)
4. label clusters by qualitative behavior (e.g., low activity, rising columns, many small blobs, few large blobs, turbulent mixing)

5. build a transition diagram between clusters

This yields a discrete macrostate model even without explicit event detection.

Workflow B: macro-features from latent trajectories
Instead of clustering, compute simple features from z(t):

- latent speed: ||z(t+1) - z(t)|| as an “activity” measure
- projection onto dominant modes: how strongly each eigenmode is expressed
- recurrence statistics: how often the system revisits similar latent regions

These features can produce an interpretable dashboard: steady state vs transition, calm vs active, eventful vs smooth.

Workflow C: event flags from prediction residuals
Train a latent evolution model and monitor prediction error over time. Spikes often align with merge/split events or lighting disturbances. You can then:

- mark event times
- treat segments between events as regimes
- fit local linear models on those segments

This turns failure into structure: the points where the model fails become the segmentation cues.

These shortcuts emphasize a philosophy: for high-dimensional, eventful systems, the most honest win is often not “a perfect forward simulator.” It is a compact representation that supports segmentation, summarization, and limited-horizon prediction while making regime boundaries visible rather than hiding them under a global average.

In the next section, we will formalize how to evaluate these claims across both exemplars. The evaluation must be strong enough to detect when a model is merely smoothing chaos into something pretty, and strong enough to reward models that preserve the right geometry, regimes, and failure horizons.

6. Evaluation: How Not to Fool Yourself

Evaluation is the difference between science and story. Learned-coordinate methods are especially prone to producing plausible-looking outputs that hide failure under smoothness, compression, or selective visualization. A model can “look right” for a few seconds and be

fundamentally wrong. It can reconstruct perfectly and still fail dynamically. It can appear stable because it has learned to damp away instability. It can appear chaotic because the decoder is a chaos renderer while the latent evolution is a simple attractor. None of these failure modes are rare. They are the default if evaluation is weak.

This section proposes an evaluation discipline designed to prevent the author from flattering themselves. The theme is adversarial humility: every metric is paired with a failure mode it is meant to catch, and every success claim is tied to tests that are harder than the demo.

We evaluate two distinct questions:

1. Forecasting: does the learned model produce accurate roll-outs for a meaningful horizon?
2. Structure: does the model preserve the geometry and qualitative invariants of the dynamics, even when pointwise prediction becomes impossible?

Both questions matter because chaotic systems can be structurally correct while being trajectory-incorrect at long horizons. The point is not to demand the impossible; it is to demand honesty about what the model actually preserves.

6.1 Forecasting metrics: one-step accuracy vs long-horizon fidelity

One-step accuracy is easy to win and easy to overinterpret. In many time series, the next step is highly constrained by continuity, so even weak models do well. In chaotic systems, one-step accuracy can be excellent while multi-step roll-outs drift into nonsense. Therefore, forecasting evaluation must be multi-scale.

One-step error: report it, but never let it be the headline. For the pendulum, one-step error can be measured in angle/velocity units (or their sine/cosine embeddings). For video, it can be pixel-wise error, but pixel error is often misleading because small spatial shifts produce large pixel differences.

Multi-step roll-out error: roll the model forward without feeding it ground truth after the initial window. Evaluate error as a function of horizon k. A model that stays accurate for 5 steps but diverges by 50 steps is not “wrong”; it has a limited forecast horizon. The failure is pretending it has more.

Error growth curves: plot error vs horizon and summarize with a small number of scalars:

- time-to-threshold: the horizon at which error exceeds a chosen threshold
- area-under-curve: integrated error over a fixed horizon
- slope of growth: how rapidly error accelerates

Distributional forecasting: for chaotic regimes, evaluate whether roll-outs preserve statistics rather than exact trajectories:

- marginal distributions of angles and velocities
- energy proxy distributions (for pendulum)
- latent “activity” distributions (for lava lamp)

Ensemble forecasting: because sensitivity amplifies initial uncertainty, it can be more honest to evaluate the model as an ensemble generator. Perturb the initial condition slightly and see whether the spread of trajectories resembles the spread observed in repeated trials.

The core principle: if the paper shows only one-step scores or only cherry-picked short roll-outs, the evaluation is not yet serious.

6.2 Geometry metrics: attractor structure, recurrence, and neighborhood preservation

Forecasting is not the only goal. For chaotic systems, structure can be preserved even when trajectory prediction fails. Geometry metrics ask: does the model generate trajectories that live on the same manifold-like set as the real system?

Attractor structure: compare the geometry of real and generated trajectories in a chosen representation:

- for pendulum: phase portraits (theta_1, omega_1), (theta_2, omega_2), and joint projections
- for lava lamp: latent-space projections and derived macrostates

The comparison should not be only visual. Use quantitative summaries:

- occupancy histograms in phase space
- distance-to-manifold statistics (how far roll-outs stray from the data cloud)
- comparison of principal components or learned latent distributions

Recurrence: chaotic systems often revisit neighborhoods of state space. Compute recurrence plots or recurrence rates:

- for each time point, count how often the system returns within epsilon of that state
- compare recurrence distributions between real and generated sequences
- compare recurrence timescales

Neighborhood preservation: learned embeddings claim to produce meaningful coordinates. Test whether neighbors remain neighbors:

- pick a set of states in real data
- find their nearest neighbors in measurement space and in latent space
- see whether the mapping preserves local neighborhoods
- more importantly, see whether neighborhoods evolve similarly under dynamics (do nearby states lead to similar short-horizon futures?)

If the latent space collapses neighborhoods that should remain distinct, the model may reconstruct well but lose dynamical discriminability.

A key caution: geometry metrics can be gamed by over-smoothing. A model that collapses to a narrow attractor can look geometrically "clean" while being wrong. Therefore, geometry metrics must be paired with variability checks: does the model preserve the observed diversity of behavior?

6.3 Stability metrics: perturbation tests and empirical Lyapunov behavior

Stability is the easiest thing for a model to fake: just add damping. But a chaotic system has a characteristic sensitivity profile that is neither perfectly stable nor explosively unstable in every direction. Stability evaluation must therefore distinguish between genuine dynamical sensitivity and model-imposed smoothing.

Perturbation tests: take a real initial window and create perturbed versions:

- for pendulum: add tiny perturbations to angles/velocities
- for lava lamp: perturb latent state or initial frames slightly (or use nearby frames as alternate starts)

Roll out each perturbed start and measure divergence between trajectories as a function of time. Compare divergence behavior between the learned model and the real system (estimated from repeated trials or from local neighbors in the dataset).

Empirical Lyapunov behavior: you do not need a formal theorem to estimate whether divergence is plausible. In practice:

- measure average log divergence growth over short horizons
- compare slopes between real and modeled trajectories

- check whether the model's divergence saturates too early (over-stable) or explodes too fast (numerically unstable)

Stability under noise: inject measurement noise at realistic levels and see whether the model's roll-outs become unphysical. A good model should degrade gracefully, not snap into a different attractor.

Invariant-set sanity checks: for the pendulum, energy-like quantities should remain within reasonable bounds given dissipation. For the lava lamp, global intensity and blob mass proxies should not drift arbitrarily if lighting is controlled. If the model violates basic conservation-ish constraints, it is likely modeling artifacts.

The principle: do not claim "stability analysis" unless you have explicitly tested whether the model is stable for the right reasons.

6.4 Baselines: naive persistence, local linear, and standard neural sequence models

A learned-coordinate method should not be evaluated against nothing. It must beat credible baselines or at least clarify what it gains.

Naive persistence: predict that the next state equals the current state (or that the next frame equals the current frame). This is surprisingly strong for high-frame-rate video, and it sets a floor for one-step prediction.

Linear autoregression in measurement space: fit a linear model directly on the measured variables or time-delay vectors. This tests whether the neural encoder is actually adding value.

Local linear models: build a library of local linearizations in state space or latent space, chosen by nearest neighbors. These often perform well in chaotic systems for short horizons and expose whether "global linearity" is necessary.

Standard neural sequence models: compare to recurrent or transformer-style predictors trained directly on the measurements. These models can be strong forecasters but often produce less analyzable representations. The comparison clarifies tradeoffs: predictability vs interpretability and stability diagnostics.

For video, add simple learned video predictors as baselines, but interpret results carefully: a video predictor can produce sharp-looking frames while drifting physically.

The key is narrative discipline: if your method performs similarly to a simple baseline, the contribution may be interpretability or stability tooling, not raw accuracy. Say that plainly.

6.5 Ablations: removing delays, reducing latent size, weakening constraints

Ablations are the strongest antidote to self-deception because they force you to show what actually matters.

Remove time delays: train the model on single frames (or single measurement vectors) and compare. If performance collapses, that supports the claim that delays are doing essential state reconstruction.

Reduce latent dimension: sweep latent sizes and report the tradeoff between reconstruction quality, forecast horizon, and structural fidelity. If a very small latent performs well, that is evidence of compressibility. If performance saturates at a higher latent dimension, that is evidence against the “one equation” narrative.

Weaken dynamical constraints: train a pure autoencoder without latent linear evolution and show that reconstruction is not enough. Then gradually increase dynamical constraint strength and show where spurious stability appears. This reveals whether the method is genuinely learning dynamics or merely compressing.

Shorten vs lengthen training horizon: if long-horizon training produces artificial damping, you need to report that and adjust objectives. If long-horizon training improves structural fidelity without damping, that is a real win.

For video, test sensitivity to augmentations: small brightness shifts, slight camera jitter, background changes. If embeddings are meaningful, they should be relatively invariant to nuisances.

Ablations do not exist to impress. They exist to catch the simplest and most embarrassing alternative explanations.

6.6 Out-of-distribution tests: new energy levels, new temperatures, new lighting

The most reliable way to detect story is to change the world slightly and see whether the model breaks in a way that reveals what it learned.

Double pendulum OOD tests:

- new energy levels: different release heights or initial velocities
- parameter shifts: slightly different friction, added small mass, changed arm length
- unseen initial-condition regions: start angles not present in training

A model that has learned a state-like representation should degrade gracefully. A model that has memorized trajectories will fail abruptly.

Lava lamp OOD tests:

- new temperatures: different warm-up states or different lamp power settings
- new lighting: controlled brightness shifts, changed ambient lighting color temperature
- camera changes: slight viewpoint shift, different camera sensor, compression settings

These tests separate “lamp dynamics” from “camera dynamics.” If the model fails under lighting changes, it may have learned the rendering rather than the physical behavior.

Cross-session generalization: the hardest and most honest test is to train on one recording session and evaluate on another day. If performance collapses, that is valuable information: the system is nonstationary, the measurement conditions are too variable, or the representation is not invariant enough.

Finally, report failure explicitly. Out-of-distribution testing is not a checkbox. It is the point at which the paper proves it is not a hype document. A model that works only in a narrow regime can still be useful, but only if its regime is declared and its boundaries are demonstrated.

This evaluation framework is intentionally demanding. It is designed not to maximize reported performance, but to maximize epistemic integrity. The next sections will build on this discipline: interpreting what learned modes mean, identifying when linear latent dynamics are genuinely informative, and specifying a reusable protocol that future readers can apply to other systems without inheriting the “one equation” myth.

7. Interpretation: What the Learned Linear Model “Means”

A learned linear latent model tempts the mind in two opposite directions. One temptation is cynicism: “it’s just a neural network, so nothing means anything.” The other temptation is metaphysics: “the model discovered the hidden equation of the universe.” Both are mistakes. The correct stance is instrumental and testable: the learned model defines a coordinate system in which evolution is simpler, and the linear operator in that coordinate system has a spectral structure that can be interpreted as modes, timescales, and directions of growth or decay. Whether those interpretations correspond to anything physically meaningful depends on validation, stability under perturbations, and the ability to map modes back to measurement space in a reproducible way.

Interpretation here is not decoration. It is the entire point of insisting on latent linearity. If we only wanted prediction, we could use black-box sequence models. We pursue linear latent dynamics because linear systems are analyzable and their analysis produces objects we can

reason about: eigenmodes, invariant subspaces, and stability signatures. The question is what those objects mean, and how not to lie about them.

7.1 Eigenmodes as reusable features and coordinate axes

If latent evolution is approximately linear, then the operator governing evolution (call it A in discrete time or G in continuous time) can be analyzed spectrally. In plain language, the spectrum tells you how the system decomposes into patterns that evolve independently under the linear approximation.

Eigenmodes serve three practical roles.

As coordinate axes: In a good latent representation, the eigenvectors of the operator can become “natural directions” in which the system’s behavior unfolds. This can be more meaningful than the raw latent coordinates, which might be arbitrary up to rotation. Spectral coordinates can give you a stable basis for comparison across training runs.

As timescales: Eigenvalues encode growth/decay rates and oscillation frequencies. Even if the system is nonlinear, these rates can provide a useful summary: which patterns decay quickly, which persist, which oscillate, and which are unstable. In the double pendulum, this may correspond to dominant oscillatory behaviors at certain energy levels, or to slow decay due to friction. In the lava lamp, it may correspond to slow heating cycles, global circulation patterns, or fluctuations in “activity” level.

As reusable features: Once you have an eigenmode decomposition, you can use mode amplitudes as features for downstream tasks: regime classification, anomaly detection, clustering of behaviors, or indexing segments of video. This is where interpretation becomes operational. If mode amplitudes predict regime changes or correlate with physical interventions (changing pendulum energy, changing lamp temperature), then the modes are not merely mathematical artifacts; they are useful coordinates.

A critical discipline: eigenmodes must be tested for stability under retraining. If the model is retrained with different random seeds and produces wildly different modes, interpretation is fragile. The correct move is to look for mode families that recur up to sign and permutation, or to align modes across runs using simple matching criteria. Reproducible modes are the first sign that “meaning” is not being invented by the analyst.

7.2 Pseudo-eigenfunctions in mixing/chaotic regimes

Strongly chaotic systems often behave like “mixing” processes: trajectories explore state space in a way that does not settle into simple periodic patterns. In such regimes, the classical hope of capturing the system with a small set of clean eigenfunctions can fail. This is not a weakness of the learning method; it can be a property of the dynamics.

This is where pseudo-eigenfunctions matter. A pseudo-eigenfunction is a function that behaves like an eigenfunction over a finite time horizon or within a restricted region of state space, even if no globally valid finite eigenfunction basis exists. In practice, learned models often discover coordinates that are locally coherent: they evolve approximately linearly for a while, then lose coherence as mixing dominates.

This is precisely why “one equation” narratives are misleading. A finite-dimensional linear model can be a good approximation for certain behaviors, energies, or time horizons without being a global law. In the double pendulum, you may get good linear-latent behavior for moderately energetic motion and poorer behavior in fully tumbling regimes. In the lava lamp, you may get coherent modes during steady bubbling and lose coherence during event-heavy merge/split periods.

Pseudo-eigenfunction interpretation demands humility. The correct question is not “is this a true eigenfunction of the system?” but “over what horizon and under what conditions does this coordinate behave predictably as if it were an eigenfunction?” That reframing turns interpretation into a measurable statement and prevents it from becoming metaphysical.

7.3 When interpretability is real vs when it is a story

Interpretability is real when it is constrained by tests the author cannot fake easily. It becomes a story when it is based on retrospective pattern-matching and selective visualization.

Here are practical criteria for real interpretability.

Reproducibility: the same qualitative modes appear across retraining runs and across data subsets. If interpretability depends on one lucky run, it is not interpretability; it is coincidence.

Intervention alignment: when you change a physical control parameter, the latent features change in a consistent, monotonic, or otherwise explainable way. For the pendulum, increasing energy should shift mode amplitudes in predictable directions. For the lava lamp, raising temperature should change activity-related features and regime occupancy. If latent variables are meaningful, they should respond to interventions more cleanly than raw pixels.

Predictive utility beyond reconstruction: a latent coordinate is not “meaningful” because it reconstructs data. It is meaningful if it predicts something you care about: regime changes, event likelihood, forecast horizon, or stability classification. If a latent variable has no predictive utility, attaching a physical label to it is a story.

Robustness to nuisances: for the lava lamp, if “meaningful modes” change drastically under modest lighting shifts, then the mode is probably encoding lighting rather than lamp physics. Interpretability must survive nuisance variation.

Out-of-distribution persistence: if a mode remains coherent when the system is pushed slightly outside the training conditions, it is more likely to reflect real structure. If it collapses instantly, it may be a fitted artifact.

Conversely, here are signs interpretation has slipped into story.

Decoder illusion: the outputs look rich and plausible, but the latent evolution is simple and over-stable. The analyst then reads meaning into the output textures, not into the latent dynamics.

Label-first reasoning: the analyst decides what the mode “should” mean and then finds a plot that supports it. Real interpretation runs the other way: modes emerge, then are tested against interventions and metrics.

Overclaiming ontology: saying “this mode is the true hidden variable” or “this is the law of chaos.” A learned mode can be a useful coordinate without being ontologically privileged.

The discipline is to treat interpretation as a hypothesis subjected to stress tests, not as a narrative reward for having trained a model.

7.4 Linking latent variables to physical quantities: cautious mapping strategies

Mapping latent variables to physical quantities is the most tempting step and the most error-prone. But it is also where the method becomes genuinely valuable if done carefully. The right stance is cautious triangulation: multiple weak indicators that agree are better than one strong-sounding label.

Here are practical mapping strategies that are honest.

Correlation with measured quantities: start with what you actually measure. For the pendulum, correlate latent coordinates and mode amplitudes with angles, velocities, energy proxies, and dissipation estimates. Look for stable correlations across runs and across time segments. If a latent coordinate correlates with an energy proxy in a stable way, it may be energy-like.

Sensitivity to controlled parameters: deliberately vary one parameter at a time and observe latent changes. For the pendulum, vary initial energy and friction (for example by changing pivot tension). For the lamp, vary temperature (warm-up state) or lamp power and lighting separately. If a latent coordinate responds to temperature but not lighting, it is more likely to be physical.

Reconstruction gradients: ask what changes in the reconstructed measurement correspond to movement along one latent direction. For video, move z slightly along one eigenvector direction, decode, and observe what changes in the frame. If moving along a mode consistently

increases blob count, or shifts mass upward, that is evidence the mode controls that macro-feature. The key is to do this systematically, not by cherry-picking.

Macro-feature alignment: compute hand-engineered macro-features (blob count, centroid height distribution, total wax area) and see whether they align with latent coordinates. The purpose is not to replace learned features with hand features, but to ground interpretation. If macro-features align with a latent axis, you have a bridge.

Local linear probes: fit simple linear regressors from latent coordinates to physical quantities using held-out data. If a physical quantity is linearly predictable from z across sessions, z likely encodes it robustly. If prediction works only on training data, interpretation is suspect.

Mapping must also respect non-uniqueness. Latent spaces are often identifiable only up to rotation and scaling. Two different training runs can yield different coordinates that represent the same underlying subspace. Therefore, mapping should often be done at the subspace or mode-family level rather than at the individual coordinate level.

Finally, the paper should treat all mappings as provisional. The honest language is: “this latent direction behaves like an energy proxy under these tests,” not “this latent variable is energy.” This keeps interpretation scientific: a collection of falsifiable claims about correlations and intervention responses, rather than a metaphysical assignment of essence.

In summary, the learned linear model “means” something when it yields reproducible modes, when those modes correspond to stable timescales and regime structure, and when cautious mapping links them to physical interventions and macro-features. It means nothing when it is interpreted purely by aesthetic inspection of reconstructions or by post-hoc labeling without stress tests. The rest of the paper will build on this by identifying limits and risks explicitly, and by proposing a reusable protocol that forces interpretability to earn its keep.

8. What This Enables Beyond These Two Toys

The double pendulum and the lava lamp are “toy” systems only in the sense that they are accessible and visually intuitive. The method they stress-test is not toy at all: learning a coordinate system in which complex time series evolve approximately linearly is a general strategy for turning raw measurements into analyzable dynamics. If it works, it gives you three capabilities that are difficult to get simultaneously with black-box predictors:

1. usable forecasting over a meaningful horizon
2. a compact state representation that can be stored, compared, clustered, and monitored
3. linear-systems tools for modes, stability signatures, and regime structure

The value of these capabilities is not that they replace domain knowledge. The value is that they provide an “analysis layer” that can be attached to sensor streams even when first-principles modeling is unavailable, too expensive, or too brittle. The cost is that evaluation must remain strict: the method can be made to look impressive in nearly any domain if you allow storytelling. This section describes what becomes possible when the method is treated as an instrument rather than a miracle.

8.1 Sensors and machines: vibration, bearings, and fault precursors

Industrial machines produce time series that are noisy, partially observed, and often nonstationary. Vibration sensors, acoustic sensors, current draw monitors, and temperature probes can provide rich streams, but the underlying system is complicated: friction changes, wear accumulates, loads vary, and the operating regime shifts.

In this setting, learned-coordinate linearization can help in several concrete ways.

Latent health coordinates: a latent embedding can compress a high-dimensional sensor stream into a small set of coordinates that track “where the machine is” in its behavior space. This can become a health manifold: normal operation occupies a region; degradation moves the system along a trajectory.

Mode-based diagnostics: linear latent evolution yields modes with characteristic frequencies and decay/growth rates. Many machine faults manifest as changes in oscillatory content: new harmonics, growth of particular bands, or changes in damping. If the latent modes are stable and reproducible, changes in their amplitudes can become early-warning signals.

Anomaly detection with geometry: rather than declaring “anomaly” when a scalar threshold is crossed, you can declare anomaly when the system’s latent trajectory exits its learned normal region or when recurrence statistics change. This reduces false alarms driven by benign load variation and focuses attention on structural deviations.

Forecasting of failure precursors: in some cases the goal is not to predict the full sensor stream, but to predict the approach to unstable behavior: chatter in machining, resonance build-up, bearing failure onset. A latent linear model can forecast the growth of an unstable mode even when raw time-domain prediction is messy.

The nontrivial discipline is out-of-distribution testing: machines experience loads and environments not present in training data. Therefore, evaluation must explicitly vary operating conditions. If the latent model merely learns “load signatures” and labels them as faults, it will produce confident nonsense.

8.2 Circuits and power systems: oscillations, instabilities, regime shifts

Electrical and power systems are full of dynamics that are well-described in principle but difficult in practice because they are networked, nonlinear, and subject to switching and control loops. The measurement streams (voltages, currents, frequency, phase angles, harmonics) can be high dimensional across many nodes.

Learned-coordinate linearization is a natural fit here because many power-system questions are already posed in linearized terms: stability margins, oscillation modes, damping ratios, and small-signal stability.

Oscillation mode extraction: if the latent operator yields eigenmodes with clear oscillatory structure, those modes can serve as automatically extracted oscillation patterns. In power grids, the ability to detect inter-area oscillations and damping changes is valuable.

Regime shift detection: switching events, protection trips, and controller mode changes create distinct regimes. A single global linear model often fails; a multi-regime latent model can identify when a regime changes and fit local linear operators per regime.

System identification from measured data: in systems where schematics and parameter values are uncertain or proprietary, learning a state representation and a linear latent evolution provides a data-driven surrogate model that can be used for simulation and stress testing.

The key risk is confusing control artifacts for system physics. A latent model trained on data from a particular controller configuration may “learn the controller,” not the plant. Therefore, evaluation should include controller changes and injection tests (where permissible) to see whether latent coordinates track plant dynamics or policy dynamics.

8.3 Physiology and neuroscience: reduced dynamics without oversimplification

Biological time series are notorious: noisy, heterogeneous, and influenced by unobserved variables. Yet they often display structured dynamics: rhythms, attractor-like behavior, and regime changes (sleep stages, seizure onset, autonomic shifts).

The promise of learned-coordinate linearization in physiology and neuroscience is not that biology becomes linear. The promise is that a compact latent representation may reveal stable timescales and modes that are useful for classification, prediction, and interpretability.

State representations for complex signals: EEG, ECG, respiration, pupil dynamics, and multi-sensor wearables can be embedded into a latent state that captures the system’s current regime more robustly than any single scalar feature.

Mode-based summaries: physiological systems exhibit oscillations (heart rate variability bands, respiratory rhythms, neural rhythms). A latent linear model can provide mode amplitudes that act as robust features for detecting transitions and anomalies.

Event prediction with humility: the most dangerous mistake is claiming strong deterministic prediction for events like seizures or arrhythmias without accounting for uncertainty. The disciplined use is to predict risk proxies and early-warning signatures (increasing variance, changing recurrence, mode growth) with explicit failure horizons.

Reduced dynamics without erasing heterogeneity: one advantage of latent embeddings is that they can preserve individual differences while still enabling common mode structure. That can support personalization without collapsing into storytelling.

The ethical discipline here is strong: claims must be bounded, evaluation must be out-of-distribution across subjects, and uncertainty must be preserved. In high-stakes biology, the model should be treated as a research instrument, not as a diagnostic oracle.

8.4 Climate and geophysical data: compression with accountability

Climate and geophysical systems are where “chaos has a formula” rhetoric becomes especially toxic, because the public is already primed for oversimplification. Yet these are also domains where learned coordinate systems can be genuinely valuable as analysis tools.

Spatiotemporal compression: large gridded fields (temperature, pressure, wind, ocean height) can be compressed into latent coordinates that capture dominant patterns. This is not new in spirit (think of classical decomposition methods), but representation learning can capture nonlinear relationships and state-dependent patterns.

Forecasting and nowcasting: short-horizon prediction tasks (storm evolution, precipitation nowcasting) can benefit from latent dynamics models. The key is to evaluate across seasons and regions, and to separate “visual plausibility” from meteorological correctness.

Regime detection: many geophysical phenomena have regime structure (blocking patterns, oscillation phases, circulation shifts). Latent mode amplitudes can provide compact indicators of regime occupancy and transitions.

Accountability and failure boundaries: perhaps the most important contribution is not improved forecast accuracy, but improved clarity about when the learned representation stops being valid. A latent linear model that produces explicit prediction residuals and mode breakdown can serve as a warning system: it tells you when the system is entering behavior not present in the training distribution.

The discipline is to treat compression as a representation for analysis, not as an announcement that geophysics is “solved.” The method can help with structure discovery and summarization, but it does not eliminate irreducible uncertainty or hidden drivers.

8.5 Control prospects: using latent linearity for planning and stabilization

The most direct payoff of linear latent dynamics is control. Control theory has powerful tools for linear systems: planning, stabilization, estimation, and robust control. If a nonlinear system can be represented as approximately linear in a learned latent space, then these tools can be applied more directly.

Latent state estimation: the encoder can function as a state estimator, mapping observations into a latent state suitable for prediction. This is valuable when sensors are noisy and the physical state is not directly measurable.

Planning in latent space: if latent evolution is linear (or close), planning reduces to a tractable optimization problem. You can forecast the effect of control inputs in latent space and choose actions that steer the system toward desired regions.

Stabilization by mode suppression: if certain modes correspond to unstable behavior (growth directions), control can target reducing their amplitudes. In machines, this might suppress resonance. In power systems, it might increase damping. In robotics, it might reduce oscillations.

Hybrid control for eventful systems: the lava lamp teaches that some systems require multi-regime or event-based descriptions. Control in such settings is inherently hybrid: continuous control within regimes, discrete logic at transitions. Learned embeddings can help by providing a regime indicator and a latent state for each regime’s controller.

The central caution is the same: control is where self-deception becomes dangerous. If the latent model invents stability or hides instability, a controller built on it can fail catastrophically. Therefore, control-oriented use demands the strongest evaluation: perturbation tests, out-of-distribution testing, and explicit safety margins.

In summary, what these “two toys” enable is not a viral headline but a reusable instrument: a way to discover analyzable coordinates from complex time series, extract modes and timescales, detect regimes and anomalies, and potentially design controllers that operate on learned state representations. The method is powerful precisely because it is not domain-specific. It is also dangerous for the same reason. Its legitimate value appears only when it is paired with evaluation that refuses to confuse compression with understanding.

9. Limits, Risks, and Misreadings

Every powerful modeling idea comes with a predictable set of misreadings. Learned-coordinate linearization is no exception. In fact, it is unusually vulnerable because it produces exactly the kind of artifact that a human mind craves: a compact “law-like” object (a linear operator) that feels like understanding. The danger is not only public hype. The deeper danger is self-hype: the researcher convinces themselves that a neat latent equation implies the world is neat.

This section is not an apology for the method. It is a set of boundary conditions that prevent the paper from becoming the thing it criticizes. The central theme is that coordinate changes can simplify representation, but they do not change the underlying epistemic situation. Chaos does not become tame simply because you found a tidy latent space; you may simply have moved complexity into the encoder/decoder and into the assumptions that your training data silently enforces.

9.1 Determinism is not simplicity

The first and most common misreading is the slide from “deterministic” to “simple.” Determinism means that the system’s evolution is rule-governed given a full state and inputs. It does not mean that the rule is low-dimensional, easily compressible, or human-legible.

A double pendulum is deterministic, but practical prediction is limited by sensitivity. The fact that the system is low-DOF does not guarantee that it admits a globally accurate low-dimensional linear surrogate. The best you may get is a useful approximation over a finite horizon or within a restricted regime.

A lava lamp can be deterministic at the level of fluid mechanics, but the observed behavior is the output of a high-dimensional field interacting with boundaries and temperature gradients. Even if the underlying equations are lawful, the state space is large and the observation channel is lossy. “Deterministic” here is almost irrelevant to whether your learned embedding will be small and stable.

This is the correct mental model: determinism tells you that the world is not making decisions. It does not tell you that the world is compressible.

9.2 Stochastic drivers and hidden variables: when embedding can’t rescue you

There are situations where no coordinate trick can save you, because the problem is not representation but missing information.

Hidden variables: if the evolution depends on unobserved variables that do not leave a signature in the measurements, you cannot reconstruct them from data alone. A learned embedding can only represent what the observation stream contains. If the driver is external

and unmeasured, the best you can do is a conditional model: “given what I can see, here is a distribution over futures,” not a deterministic forecast.

Stochastic drivers: many real systems are driven by noise or by effectively random inputs (from the model’s perspective). In such cases, a latent linear model may still summarize structure, but it cannot produce long-horizon point forecasts. The correct output is a probabilistic envelope, not a single trajectory.

Nonstationary parameters: if system parameters drift (friction changes, lamp temperature changes, sensor calibration shifts), you are not modeling one system but a family of systems. A learned embedding may adapt by adding latent dimensions that encode drift, or it may fail by inventing spurious damping. Either way, the “one equation” story collapses because the equation is not fixed.

The disciplined conclusion is simple: if you do not measure the inputs and you cannot infer them from the outputs, there is no state representation that will make the future uniquely predictable. Embeddings can compress what you see. They cannot conjure what you did not measure.

9.3 Topological change and non-smooth events

Some of the hardest failures arise not from chaos but from discontinuities in the representation.

Topological change: the lava lamp’s merge/split events are the canonical example. If your representation depends on object identity, you will face discontinuities when identity changes. A latent linear flow assumes smooth evolution in a fixed-dimensional coordinate space. Merge/split can violate that assumption at the perceptual level, even when the underlying physics is smooth.

Non-smooth events: impacts, stick-slip friction, contact events, and saturation in actuators create real discontinuities or sharp nonlinearities. Many mechanical systems in the real world are not smooth ODEs; they are hybrid systems. A learned linear latent model can approximate them within regimes but will often smear transitions unless it explicitly models regime switching.

Event aliasing: even if events are real, poor sampling can turn smooth transitions into apparent jumps. The model then learns to treat aliasing artifacts as dynamics. This is a measurement problem masquerading as a modeling problem.

The correct posture is to treat these systems as hybrid by default: smooth dynamics plus events. A single global linear latent operator is often the wrong abstraction. A mixture of operators, a switching model, or an event-conditioned model is more faithful.

9.4 Dataset temptation: the difference between fit and understanding

A learned model is an optimizer. It will exploit any shortcut that reduces loss, including shortcuts that destroy interpretability.

Camera shortcuts: in video, the model can learn lighting changes, compression patterns, or background cues. It can "predict" the camera pipeline rather than the physical system.

Trajectory memorization: for the pendulum, a model can learn a narrow distribution of initial conditions and produce plausible roll-outs for those conditions while failing elsewhere. If evaluation does not include diverse starts and out-of-distribution regimes, the model will appear far stronger than it is.

Decoder hiding: the model can keep latent dynamics simple and push complexity into the decoder, producing outputs that look rich while the latent flow is trivial. This is especially seductive because it produces good-looking videos and plots.

Loss-function gaming: multi-step prediction objectives can be minimized by damping. The model can converge to a stable attractor because stable attractors reduce long-horizon error. This produces the illusion of "stability discovery" even when the system is not stable at that regime.

The essential warning is this: fit is easy. Understanding is the ability to survive adversarial tests. If the paper does not report failures, ablations, and out-of-distribution behavior, it is describing fit, not understanding.

9.5 The rhetoric hazard: "chaos has a formula" as category error

Finally, the viral headline itself is a category error. It confuses a representational convenience with an ontological discovery.

A latent linear model is a tool. It is a way to write down a compact surrogate evolution in coordinates that were learned to make that surrogate plausible. The existence of such a surrogate does not imply that chaos is "secretly simple" in the original variables, nor that prediction becomes easy, nor that a single human-legible law has been found.

The "formula" claim is also misleading because it hides where complexity went:

- into the learned encoder and decoder
- into the choice of delays, horizons, and training objectives
- into the selection of data regimes and measurement conditions
- into the implicit assumptions that the training distribution encodes

The disciplined language is therefore different. It says: “We can sometimes learn coordinates in which a compact linear operator provides a useful approximation over defined horizons and regimes, enabling analysis and forecast tools that are otherwise difficult to apply.” That is a strong claim. It is also falsifiable. It does not require mythology.

This paper’s goal is to rescue the legitimate idea from its rhetoric. Chaos does not become a napkin equation. But representation can become disciplined, compact, and analyzable—if and only if we keep the limits explicit, evaluate aggressively, and refuse to confuse an elegant surrogate with the world itself.

10. A Reusable Protocol

This paper’s core claim is practical: learned-coordinate linearization can be treated as an instrument for analyzing complex dynamics, but only if the workflow is disciplined enough to prevent self-deception. The reader should be able to take the method and apply it to a new system—another mechanical apparatus, a different fluid visualization, a biological sensor stream—and obtain not only a model, but a credible statement of what the model does, where it works, and where it fails.

A reusable protocol therefore has four components:

1. data collection standards that make “state” more recoverable and nuisance variation more controllable
2. a training recipe that reduces the likelihood of trivial solutions (especially spurious stability)
3. reporting standards that make it difficult to hide failure behind cherry-picked demos
4. a minimal benchmark suite so results can be compared across projects and over time

The protocol is intentionally lightweight. It is not a demand for lab-grade infrastructure. It is a demand for epistemic hygiene.

10.1 Data collection checklists for low-DOF vs video systems

The biggest determinant of success is often not architecture; it is measurement discipline. The checklist below separates two common worlds: low-dimensional physical systems with direct sensors, and high-dimensional visual systems captured by video.

Low-DOF systems (e.g., double pendulum, oscillators, mechanical rigs)

- State coverage: collect trajectories across a wide range of initial conditions (energies, phases, release angles). Do not train on one “pretty” regime.
- Sampling rate: sample fast enough that dynamics are resolved; avoid aliasing. If your sampling rate is limited, explicitly declare that you are modeling a coarse-grained system.
- Timestamp integrity: record accurate timestamps; handle dropped samples; resample to a uniform grid if necessary.
- Sensor calibration: calibrate angle/position sensors; document units; check for bias drift across the recording.
- Velocity estimation: if velocities are derived, use a smoothing-and-derivative method that minimizes noise amplification and phase lag. If velocities are unreliable, plan to use time-delay vectors to infer them.
- Parameter logging: record lengths, masses, pivot friction settings, and any changes during experiments.
- Repeat trials: perform repeated runs from “nominally identical” initial conditions to estimate sensitivity and measurement noise. This becomes a ground truth for divergence behavior.
- Regime labeling (optional but useful): mark segments with obvious regime differences (e.g., low-energy oscillation vs tumbling) to support later regime testing.

Video systems (e.g., lava lamp, fluids, crowds, weather visualization)

- Camera stability: fixed tripod; fixed focus; fixed viewpoint; avoid hand-held drift.
- Exposure discipline: lock exposure/white balance if possible. If not possible, record metadata and include deliberate lighting shifts as explicit OOD tests.
- Lighting control: constant lighting, no flicker; avoid changing shadows and reflections; isolate the background.
- Frame rate and timing: record a stable frame rate and timestamps; avoid variable rate capture when possible; resample carefully when not.
- Compression control: avoid aggressive compression; keep consistent codec settings across sessions when possible.

- Session structure: record long sequences that include warm-up/transient and steady-state. Do not train only on steady-state unless that is your declared scope.
- Perturbations for evaluation: deliberately include controlled perturbations (small viewpoint shift, small brightness change) in a held-out test set to measure nuisance sensitivity.
- Auxiliary sensors (optional): for systems like a lava lamp, record temperature near the base or power setting. Even a crude scalar can help interpret regimes.

The common principle: make the observation stream as informative and as consistent as is reasonable, while intentionally reserving some variations for later out-of-distribution evaluation.

10.2 Training recipe: staged objectives and horizon growth

Chaotic systems punish naive training. A model that is forced into long-horizon prediction too early often learns the easiest cheat: damping. A model that trains only for one-step prediction often looks good and then drifts during roll-out. The training recipe must therefore be staged.

Stage 0: preprocessing and representation choices

- Normalize units consistently.
- For angular variables, avoid discontinuities at +/-pi (use unwrap or sine/cosine encoding).
- For video, consider background subtraction or simple normalization, but avoid brittle segmentation as the default.
- Build time-delay vectors or short frame stacks to provide motion context.

Stage 1: reconstruction pretraining (autoencoder)

- Train encoder/decoder to reconstruct observations from the latent.
- Use a latent dimension large enough that reconstruction is not artificially bottlenecked in the beginning.
- Verify that reconstruction quality is stable across the dataset (not only on a subset of regimes).

Stage 2: introduce latent linear evolution (one-step)

- Add a linear operator in latent space and train for one-step prediction in latent and/or decoded space.
- Begin with a modest weight on the dynamical term so the latent does not collapse.

Stage 3: horizon growth (multi-step roll-out in training)

- Increase prediction horizon gradually: for example 1 -> 5 -> 10 -> 25 steps.
- At each horizon increase, monitor for spurious stability: if the model starts damping everything to reduce error, adjust the loss weighting or add constraints that penalize artificial contraction.
- Use teacher-forcing early (feeding ground truth latent) and then shift to free roll-outs (feeding predicted latent) as horizon grows. The point is to train the model under the conditions in which it will be evaluated.

Stage 4: regularization for interpretability and robustness

- Encourage latent smoothness without erasing sensitivity.
- Encourage invariance to nuisances for video (small brightness changes, small shifts) via augmentation, but keep augmentations realistic.
- If using a multi-regime approach, train regime selection or switching logic only after the latent representation is stable.

Stage 5: model selection via validation that matches claims

- Choose models based on long-horizon metrics and geometry metrics, not on one-step loss.
- Select latent dimension based on tradeoffs: reconstruction, forecast horizon, structural fidelity, and robustness.
- Always keep a strict held-out test set that includes out-of-distribution conditions.

The meta-rule: training objectives should be aligned with the paper's claims. If you claim long-horizon fidelity, you must train and validate on long horizons. If you claim regime detection, you must validate regime transitions explicitly.

10.3 Reporting standards: plots, tables, and failure cases to always include

A reusable protocol requires reporting standards that prevent cherry-picking. The following items should appear in every serious report, even if they are unflattering.

Core forecasting plots

- Error vs horizon curve (multi-step roll-out), with confidence intervals across many initial conditions.
- Time-to-threshold distribution (how long forecasts remain within a defined tolerance).

- Example roll-outs from multiple initial conditions, including at least one hard case.

Geometry and structure plots

- Phase portraits for low-DOF systems: real vs generated overlays, not just a single trajectory but occupancy density or multiple trajectories.
- Latent space projections: show real vs generated distributions in a consistent embedding.
- Recurrence summaries: recurrence rate distributions or recurrence plots for representative segments.

Stability and sensitivity plots

- Perturbation divergence curves: divergence over time for small perturbations, compared between real system (or neighbor-based estimate) and learned model.
- Evidence against spurious stability: show whether latent eigenvalues imply contraction and whether contraction matches observed dissipation.

Baseline comparison table

Include at least: naive persistence, linear autoregression on delays, local linear model, and a standard neural sequence model. Report the same metrics for all models.

Ablation table

- No delays vs delays
- smaller vs larger latent dimension
- weaker vs stronger dynamical constraint
- short-horizon training vs horizon-grown training

Failure cases (mandatory)

- At least one example of drift, mode collapse, or spurious stability, with explanation of how it was detected.
- Out-of-distribution failures: new energies or new lighting where performance drops.
- A clear statement of the declared scope: “works in these regimes, fails in these regimes.”

Interpretation artifacts

- If you interpret eigenmodes, show reproducibility across training runs (mode alignment or stability of mode families).
- If you map latent variables to physical quantities, show correlation and intervention tests, not just one illustrative plot.

The moral is simple: a paper that shows only success is not showing truth; it is showing marketing.

10.4 A minimal open benchmark suite for chaotic embeddings

A benchmark suite is not about leaderboard obsession. It is about comparability and regression testing. A minimal suite should be small enough for a reader to run, yet rich enough to reveal the key failure modes discussed in this paper.

Benchmark A: low-DOF chaotic mechanics

- Double pendulum dataset with standardized sensor streams (angles and optionally velocities) at a fixed sampling rate.
- Multiple energy regimes: low oscillation, intermediate chaotic, high tumbling.
- Repeated trials from nominally identical starts to estimate sensitivity/noise.
- Evaluation tasks:
    - multi-step forecast horizon curves
    - phase portrait fidelity
    - perturbation divergence behavior

Benchmark B: eventful video dynamics

- Lava lamp video dataset recorded under controlled conditions:
    - one long warm-up to steady-state session
    - one steady-state session
    - one session with controlled lighting shift
    - optionally one session with slight viewpoint shift
- Optional auxiliary scalar: temperature or power setting.
- Evaluation tasks:

- short-horizon video prediction in latent space
- regime clustering quality (macrostate stability)
- event detection via prediction residual spikes
- robustness to lighting changes

Benchmark C: synthetic sanity checks

- A known chaotic system with clean ground truth (simulated), to test whether the method recovers expected sensitivity profiles.
- A non-chaotic periodic system, to ensure the method does not invent chaos.
- A stochastic-driven system, to test whether the method correctly expresses uncertainty limits (or at least shows bounded forecast horizons).

Benchmark D: stress tests for self-deception

- A dataset with intentionally injected camera artifacts (auto-exposure simulation) to see whether the model learns nuisance variation.
- A "shortcut" dataset where background correlates with regime labels, to test whether embeddings are robust.

Each benchmark should include a fixed train/test split and a fixed set of required metrics, so that improvements are real improvements rather than changes in evaluation choices. The benchmark suite should be designed to make the method fail in predictable ways unless it is genuinely robust, because robust methods are the only ones worth carrying into real domains.

In short, the reusable protocol is not a set of sacred steps. It is a set of guardrails. It ensures that when you apply learned-coordinate linearization to a new chaotic or complex system, you end up with a model plus a truthful description of its scope—rather than a neat equation and an inflated story.

11. Conclusion: Order Without Hype

The phrase "chaos has a formula" feels satisfying because it offers closure. It suggests that the world's messiness was an illusion and that the right mathematical key has finally been found. This paper argues for something both more modest and more powerful: not closure, but a disciplined instrument. Learned-coordinate linearization can convert certain kinds of complex time series into a representation where analysis becomes possible and forecasting becomes

tractable for a defined horizon. The price of that power is epistemic humility: you must separate what the model actually delivers from what the headline wants you to believe.

The double pendulum and the lava lamp were chosen precisely because they resist the simplest stories. The pendulum is deterministic yet quickly unpredictable. The lava lamp is lawful at the physical level yet rendered through a lossy, nuisance-filled observation channel and punctuated by topological events. If a method can extract useful latent structure from these cases under adversarial evaluation, it deserves attention. If it cannot, the failure is not embarrassing; it is the result. It tells you where representation learning hits the boundary between compressible structure and irreducible complexity.

11.1 What we can honestly claim

We can honestly claim that complex dynamics often become more analyzable after a change of coordinates, and that modern representation learning provides practical tools for discovering such coordinates from data.

We can honestly claim that a learned latent model with approximately linear evolution can provide tangible benefits:

- improved short-horizon forecasting compared to naive baselines
- compact state-like embeddings that support clustering, indexing, and regime detection
- spectral objects (modes, timescales, growth/decay signatures) that enable analysis tools unavailable to black-box predictors

We can honestly claim that “one equation” is typically a linear operator acting on a latent state, and that the usefulness of this operator is operational: it is measured by roll-out fidelity, geometry preservation, stability behavior under perturbation, and robustness under out-of-distribution shifts.

We can honestly claim that, when evaluated with discipline, these methods can produce not only successes but well-characterized failure boundaries: where drift begins, where spurious stability appears, and where topological events break smooth approximations. That boundary knowledge is itself valuable. It turns a model into an instrument rather than a marketing artifact.

Finally, we can honestly claim that the method’s most legitimate promise is not that chaos becomes simple, but that our descriptions can become more structured. The “order” that emerges is a representational order: the ability to organize, summarize, and partially predict complex behavior without pretending to eliminate sensitivity or uncertainty.

11.2 What remains open

Several questions remain structurally open, not because the method is immature, but because the problem is deep.

Global vs local linearity: Many systems may admit useful local linearizations in learned coordinates but resist a single global operator. The open question is how to represent this faithfully without exploding complexity: mixtures of operators, switching models, or continuous operator families.

Topology and events: For systems like the lava lamp, merge/split events expose the limits of smooth latent flow. The open challenge is building hybrid models that treat events as first-class rather than as glitches—models that can predict event likelihood, represent identity changes, and maintain coherent state across transitions.

Identifiability and meaning: Latent coordinates are often non-unique up to rotations and permutations. The open problem is establishing what kinds of "interpretation" are stable: subspaces, mode families, or invariant summaries that survive retraining and domain shifts.

Uncertainty and stochasticity: Many real systems have unmeasured drivers and stochastic components. The open question is how to integrate latent linear structure with honest uncertainty modeling, so that the model outputs calibrated prediction envelopes rather than overconfident trajectories.

Benchmarking and standards: A field matures when its evaluation becomes hard to game. The open work is social and technical: shared benchmarks, reporting norms that require failure cases, and agreement about what "success" means in chaotic domains where pointwise prediction must degrade.

These open questions are not footnotes. They define the frontier between useful representation and narrative temptation.

11.3 Why this matters: analysis, prediction, and humility

This matters because modern life runs on time series. Machines vibrate, grids oscillate, bodies pulse, markets fluctuate, climates shift, and videos record phenomena too complex to model from first principles in every case. We need tools that can extract structure from these streams without collapsing into either arrogance ("we solved it") or nihilism ("it's all noise").

Learned-coordinate linearization offers a middle path. It can give us analyzable representations that support forecasting where forecasting is possible, regime detection where regimes exist, and clear boundary markers where prediction fails. In other words, it can increase our practical intelligence about complex systems.

But the deepest reason it matters is methodological. The future will be full of models that produce plausible outputs and compelling stories. The only protection against that world is disciplined evaluation and a refusal to confuse compression with understanding. The aim of this paper has been to show that you can take a seductive claim—"chaos has a formula"—and transmute it into a rigorous workflow: define state vs measurement, learn coordinates with explicit dynamical constraints, test with adversarial metrics, report failures, and interpret modes only when they survive intervention and reproducibility tests.

Order without hype is not a rhetorical stance. It is a practice. And in a world increasingly shaped by learned models, that practice is as important as the models themselves.

References

Primary source behind the viral "one equation" claim

1. Moore, S. A., Mann, B. P., Chen, B. Automated global analysis of experimental dynamics through low-dimensional linear embeddings. *npj Complexity* 2, Article 36 (2025). doi:10.1038/s44260-025-00062-y. Nature+1

Secondary summaries of the same work (useful for quick orientation)

2. Duke Pratt School of Engineering. AI "learns" to build simple equations for complex systems (news release; discusses weather/large-variable framing). (2025). Duke Pratt School of Engineering

3. ScienceDaily. AI "learns" to build simple equations for complex systems (reprint/syndication of the Duke release). (2025). ScienceDaily

Foundations: chaos, delay embedding, and "change coordinates first"

4. Takens, F. Detecting strange attractors in turbulence. In *Lecture Notes in Mathematics* 898, 366 (1981). Nature

5. Fraser, A. M. & Swinney, H. L. Independent coordinates for strange attractors from mutual information. *Phys. Rev. A* 33, 1134 (1986). Nature

6. Fraser, A. M. Information and entropy in strange attractors. *IEEE Trans. Inf. Theory* 35, 245–262 (1989). Nature

7. Abarbanel, H. D., Carroll, T., Pecora, L., Sidorowich, J. & Tsimring, L. S. Predicting physical variables in time-delay embedding. *Phys. Rev. E* 49, 1840 (1994). Nature

8. Abarbanel, H. *Analysis of Observed Chaotic Data*. (Springer, 2012). Nature

9. Lorenz, E. N. Predictability: A problem partly solved. In *Proc. Seminar on Predictability*, Vol. 1 (1996). Nature

Koopman/DMD and linear operators as the "simple equation"

10. Schmid, P. J. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* 656, 5–28 (2010). Nature

11. Williams, M. O., Kevrekidis, I. G. & Rowley, C. W. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *J. Nonlinear Sci.* 25, 1307–1346 (2015). Nature

12. Mezić, I. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dyn.* 41, 309–325 (2005). Nature

13. Mezić, I. Spectrum of the Koopman operator, spectral expansions in functional spaces, and state-space geometry. *J. Nonlinear Sci.* 30, 2091–2145 (2020). Nature

14. Mezić, I. Koopman operator, geometry, and learning of dynamical systems. *Not. Am. Math. Soc.* 68, 1087–1105 (2021). Nature

15. Otto, S. E. & Rowley, C. W. Koopman operators for estimation and control of dynamical systems. *Annu. Rev. Control, Robot., Autonomous Syst.* 4, 59–87 (2021). Nature
16. Brunton, S. L., Brunton, B. W., Proctor, J. L. & Kutz, J. N. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLOS ONE* 11, e0150171 (2016). Nature
17. Brunton, S. L., Brunton, B. W., Proctor, J. L., Kaiser, E. & Kutz, J. N. Chaos as an intermittently forced linear system. *Nat. Commun.* 8, 19 (2017). Nature
18. Arbabi, H. & Mezić, I. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator. *SIAM J. Appl. Dyn. Syst.* 16, 2096–2126 (2017). Nature

Neural / autoencoder latent dynamics (encoder → linear step → decoder)

19. Tschannen, M., Bachem, O. & Lucic, M. Recent advances in autoencoder-based representation learning. arXiv:1812.05069 (2018). Nature
20. Chen, R. T., Rubanova, Y., Bettencourt, J. & Duvenaud, D. K. Neural ordinary differential equations. *Adv. Neural Inf. Process. Syst.* 31 (2018). Nature
21. Lusch, B., Kutz, J. N. & Brunton, S. L. Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* 9, 4950 (2018). Nature
22. Takeishi, N., Kawahara, Y. & Yairi, T. Learning Koopman invariant subspaces for dynamic mode decomposition. *Adv. Neural Inf. Process. Syst.* 30 (2017). Nature
23. Yeung, E., Kundu, S. & Hodas, N. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In *ACC* (2019). Nature
24. Morton, J., Witherden, F. D. & Kochenderfer, M. J. Deep variational Koopman models: Inferring Koopman observations for uncertainty-aware dynamics modeling and control. arXiv:1902.09742 (2019). Nature
25. Chen, B. et al. Automated discovery of fundamental variables hidden in experimental data. *Nat. Comput. Sci.* 2, 433–442 (2022). Nature
26. Bakarji, J., Champion, K., Kutz, J. N. & Brunton, S. L. Discovering governing equations from partial measurements with deep delay autoencoders. arXiv:2201.05136 (2022). Nature

Time-delay + Koopman observables (useful for chaotic regimes)

27. Kamb, M., Kaiser, E., Brunton, S. L. & Kutz, J. N. Time-delay observables for Koopman: Theory and applications. *SIAM J. Appl. Dyn. Syst.* 19, 886–917 (2020). Nature+1
28. Kaiser, E., Kutz, J. N. & Brunton, S. L. Data-driven discovery of Koopman eigenfunctions for control. *Mach. Learn.: Sci. Technol.* 2, 035023 (2021). Nature

Control from learned linear structure (latent linearity → planning/MPC)

29. Peitz, S., Otto, S. E. & Rowley, C. W. Data-driven model predictive control using interpolated Koopman generators. *SIAM J. Appl. Dyn. Syst.* 19, 2162–2193 (2020). Nature

30. Korda, M. & Mezić, I. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. (arXiv / draft circulated 2016–2018). arXiv

Video-as-dynamics (for the lava lamp exemplar)

31. Huang, K., Cho, D. H. & Chen, B. Automated discovery of continuous dynamics from videos. arXiv:2410.11894 (2024). Nature

32. Horn, B. K. P. & Schunck, B. G. Determining optical flow. (Classic optical-flow formulation; 1981). ScienceDirect

33. Teed, Z. & Deng, J. RAFT: Recurrent all-pairs field transforms for optical flow. (2020). ECVA

Hybrid / switching dynamics and topological events (merge/split, regime shifts)

34. Goebel, R., Sanfelice, R. G. & Teel, A. R. *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. (Princeton Univ. Press, 2012). Hybrid Control Lab

Double pendulum (canonical chaotic benchmark)

35. Shinbrot, T., Grebogi, C., Wisdom, J. & Yorke, J. A. Chaos in a double pendulum. *Am. J. Phys.* 60, 491–497 (1992). Yorke

Weather/climate AI as "adjacent context" (high-dimensional forecasting, different philosophy)

36. Lam, R. et al. Learning skillful medium-range global weather forecasting (GraphCast). *Science* (2023). doi:10.1126/science.adi2336. Science

37. Bi, K. et al. Accurate medium-range global weather forecasting with 3D neural networks (Pangu-Weather). *Nature* 619, 533–538 (2023). Nature

38. Pathak, J. et al. FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators. arXiv:2202.11214 (2022). arXiv

Related epistemic / rhetoric framing (optional; "hype resistance" in model talk)

39. Youvan, D. C. Perception as Interface: Human and AI Misreadings of Reality. ResearchGate preprint (October 2025). doi:10.13140/RG.2.2.23412.41603. ResearchGate

40. Youvan, D. C. The Receiver Thesis: Consciousness as a Universal Signal and the Centaur as a Higher-Gain Antenna. ResearchGate preprint (December 2025). doi:10.13140/RG.2.2.27564.73606. ResearchGate

The author has published over 4,800 AI-assisted papers at: https://www.researchgate.net/profile/Douglas-Youvan/research . Google Scholar has indexed these preprints, and if you want a particular reference, the AI mode of a Google search (Gemini) has efficiently catalogued these preprints.

$Z_{t+1} = A Z_t$
0 0 9.01
5 0 0.51
2 0.75 0.61
1.6.3 0.16 1.10